

# Introduction au JTAG

## *Définition*

Conçue dès 1985, sous l'initiative des grandes entreprises d'électronique du marché d'alors (IBM, AT&T, ...), l'interface **JTAG (Joint Test Action Group)** se présentait comme la solution incontestable pour effectuer des tests, dont l'approche est universelle à tout composant logique. Cette interface, permet donc de tester chaque composant d'un ensemble de systèmes électroniques de manière isolée, et d'observer les états de toutes les lignes internes et périphériques des composants. Il est alors possible de déterminer les circuits défectueux, et les pistes en court-circuit ou altérées.

## *Les intérêts de l'interface*

- Le principal intérêt de ce procédé est de rendre possible la programmation d'un circuit par la même interface, **sans pour autant perturber ou altérer la fonctionnalité des circuits en pleine activité.**
- Un autre intérêt majeur, est **de n'avoir besoin que de 4 lignes** pour fonctionner. Ce qui réduit considérablement l'espace occupé par l'ajout du bus JTAG sur une carte, et rend la connectique facile à réaliser.
- De plus, **les tests sont réalisables quels que soient les composants utilisés et on détecte de manière sure et précise les défauts de conception.** Il devient donc possible de tester des infrastructures électroniques complexes, tout en évitant de démonter les équipements pour tester et mesurer physiquement les points critiques, d'où une économie considérable de temps et d'argent pour la main-d'œuvre.
- Enfin, **la rapidité** est un autre de ses intérêts. il est possible d'effectuer les tests suffisamment rapidement pour réagir dans un délai très court.

## *Le JTAG aujourd'hui*

La conception JTAG est aujourd'hui définie par la norme IEEE 1149.1. Elle a été standardisée par la quasi-totalité des constructeurs de la microélectronique : ST microelectronics, Texas Instruments, Xilinx, Altera, etc... Le soutien de ces grandes sociétés rend l'interface toujours plus souple et accessible.

# ETUDE THEORIQUE DE LA NORME JTAG

## 1 - Le bus JTAG

La norme JTAG est spécifiée par la convention IEEE 1149.1. L'ensemble des connexions JTAG regroupant une collection de circuits est appelé « Chaîne JTAG ». Il s'agit d'un réseau mixte série/parallèle sur lequel les composants sont greffés en cascade (*Figure 1*). Le bus est constitué de 4 lignes, dont 2 qui sont communes à tous les composants.

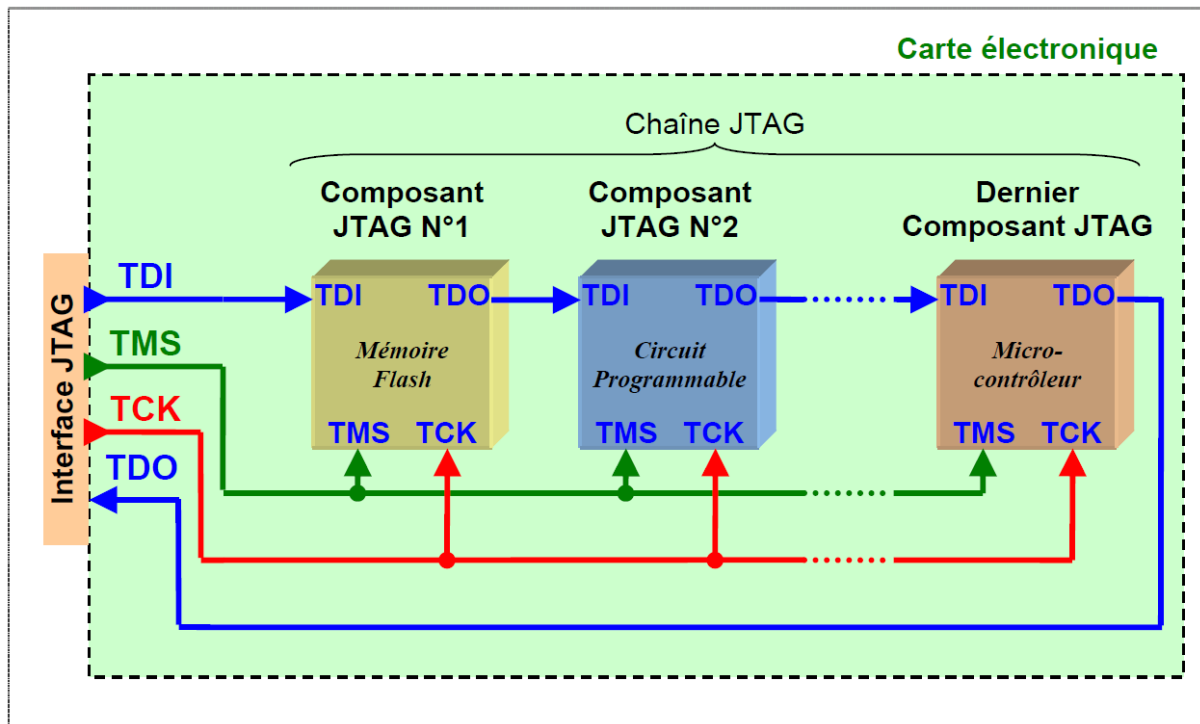


Figure 1 : Disposition d'une chaîne JTAG en aval de l'interface.

## 1 - 1 Descriptif des lignes

### • Données en série

Les données sont sérialisées pour parcourir tous les composants. Chacun des éléments JTAG dispose des deux lignes : une entrée appelée **TDI** (Test Data Input) et une sortie **TDO** (Test Data Output). Un composant récupère donc les données par TDI que son prédécesseur lui a délivré par sa ligne TDO.

Il faut remarquer que la ligne TDO se met en haute impédance après son activité pour ne pas consommer de courant.

### • Mode de fonctionnement

Pour synchroniser les activités de l'interface, on distingue des modes de fonctionnement qui sont indiqués par une ligne commune **Test Mode Select (TMS)**. Le nombre de 0 et de 1 qui sont divulgués par la ligne, permet de sélectionner le mode de fonctionnement. Le mode est validé par la *Machine JTAG*, qui constitue le protocole collectif JTAG, qui sera expliqué un peu plus loin. Suivant le mode validé, les instructions qui circulent sur les lignes TDI et TDO peuvent être soit des données, soit des instructions. La ligne TMS étant commune, tous les composants sont simultanément dans le même mode de fonctionnement.

### • Synchronisation

L'horloge apportée par la ligne **Test Clock (TCK)** permet de définir la cadence de transfert des données, qui peut être irrégulière. Ainsi, toutes les entrées (TDI et TMS) sont prises en compte uniquement sur un front montant de l'horloge, tandis que la sortie TDO est validée après un front descendant de l'horloge (Figure 2).

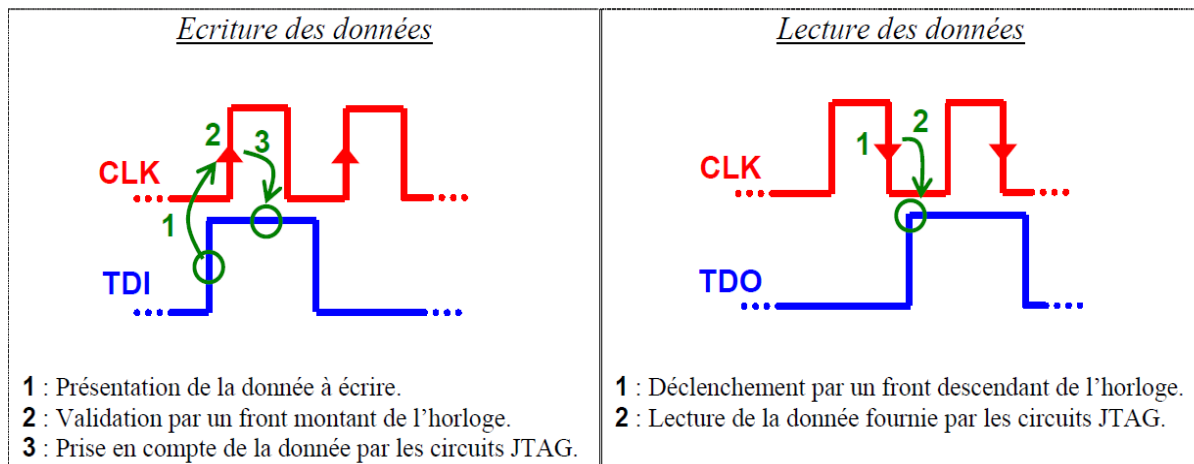


Figure 2 : Processus d'écriture et de lecture.

## 1- 2 Caractéristiques électriques

- L'interface JTAG est compatible aux technologies TTL et CMOS. Cela signifie que l'on peut les utiliser soit en logique CMOS (0 – 3,3 V), soit en logique TTL (0 – 5V). En effet, la plupart des composants logiques actuels utilisent des transistors de type CMOS ou BiCMOS.
- Concernant la vitesse de fonctionnement, le tableau suivant (Figure 3) résume les performances temporelles définies par la norme IEEE 1149.1. Elles concernent la technologie électronique actuelle et peut encore suivre des évolutions.

Portée	Paramètre temporel	Min	Max	Unité
TCK	Période de l'horloge.	100		ns
	Détection de l'état haut.	50		ns
	Détection de l'état bas.	50		ns
Circuiterie JTAG	TDI : Initialisation pour recevoir une nouvelle donnée.	20		ns
	TDI : Prise en compte de la donnée.	45		ns
	TDO : Délai de délivrance des données.		25	ns
	TDO : Délai de mise en haute impédance pour validation.		25	ns
	TDO : Délai de sortie de la haute impédance.		25	ns
Registres	TDI : Initialisation pour recevoir une nouvelle donnée.	20		ns
	TDI : Prise en compte de la donnée.	45		ns
	TDO : Délai de délivrance des données.		25	ns
	TDO : Délai de mise en haute impédance pour validation.		25	ns
	TDO : Délai de sortie de la haute impédance.		25	ns

Figure 3 : Tableau des performances temporelles de l'interface JTAG

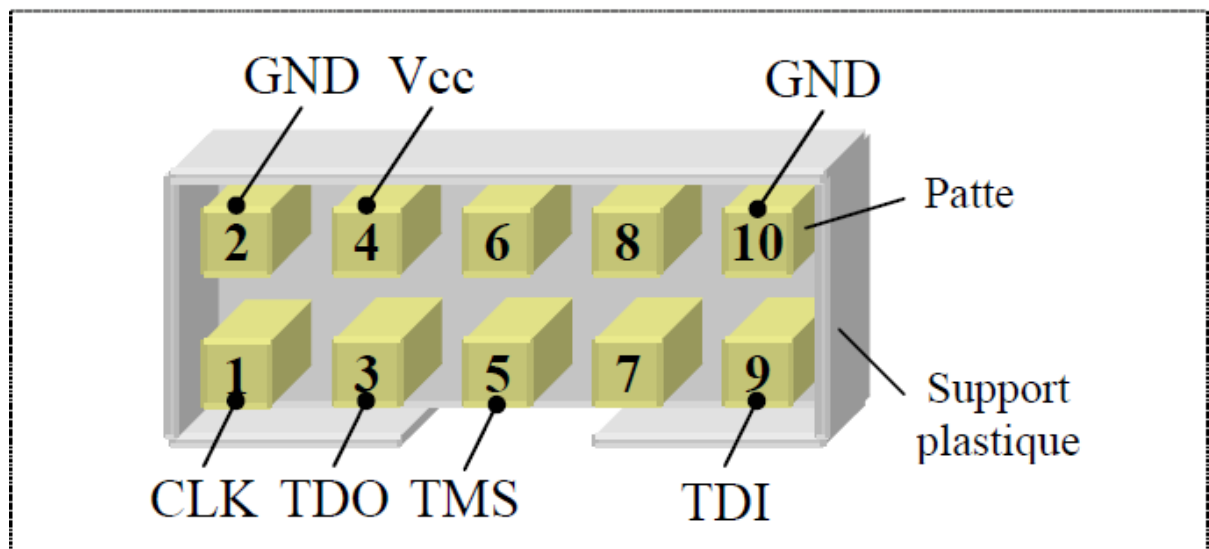
- Dans ce tableau, on peut remarquer que l'horloge de l'interface, doit avoir une fréquence qui ne dépasse pas 10 MHz, ce qui est théoriquement suffisant pour transmettre des informations d'une centaine de Koctets (programme) en quelques dixièmes de secondes.
- Par ailleurs, il faut veiller à maintenir les données transférées par TDI pendant au moins 65 ns pour qu'elles soient prises en compte. Il faut aussi que les données ne soient lues

que 25 ns après le front descendant de l'horloge.

### 1 - 3 Connecteur HE10

Un connecteur pour l'interface JTAG est très simple à réaliser puisqu'il ne faut que 4 lignes logiques. Le connecteur n'est pas standardisé, mais celui qui est le plus couramment employé est un HE10 ([Figure 4](#)).

La configuration de ce connecteur est recommandée par *Altera* pour la réalisation du câble PC/JTAG. Ce connecteur dispose donc de 10 broches qui ne sont pas toutes utilisées. Celles qui restent peuvent servir pour apporter une alimentation extérieure, mais aussi pour inclure des lignes JTAG qui sont peu utilisées (ex : TRST = Test Reset).



*Figure 4 : Connecteur HE10 mâle, côté circuit.*

## 2- Synoptique de l'interface physique

Un composant compatible JTAG est constitué d'un circuit logique joint à une interface JTAG. Le circuit de l'interface est implanté sur toute la périphérie du circuit logique. Cette interface a une structure proche d'une unité de traitement informatique : elle fonctionne avec des données et des instructions, possède des registres caractéristiques et un contrôleur aiguille les informations. Pour un composant, elle est définie par le synoptique de la [Figure 5](#).

Il est essentiel que cette interface soit parfaitement asynchrone avec le circuit qu'elle supporte ; c'est pourquoi elle dispose d'une horloge propre.

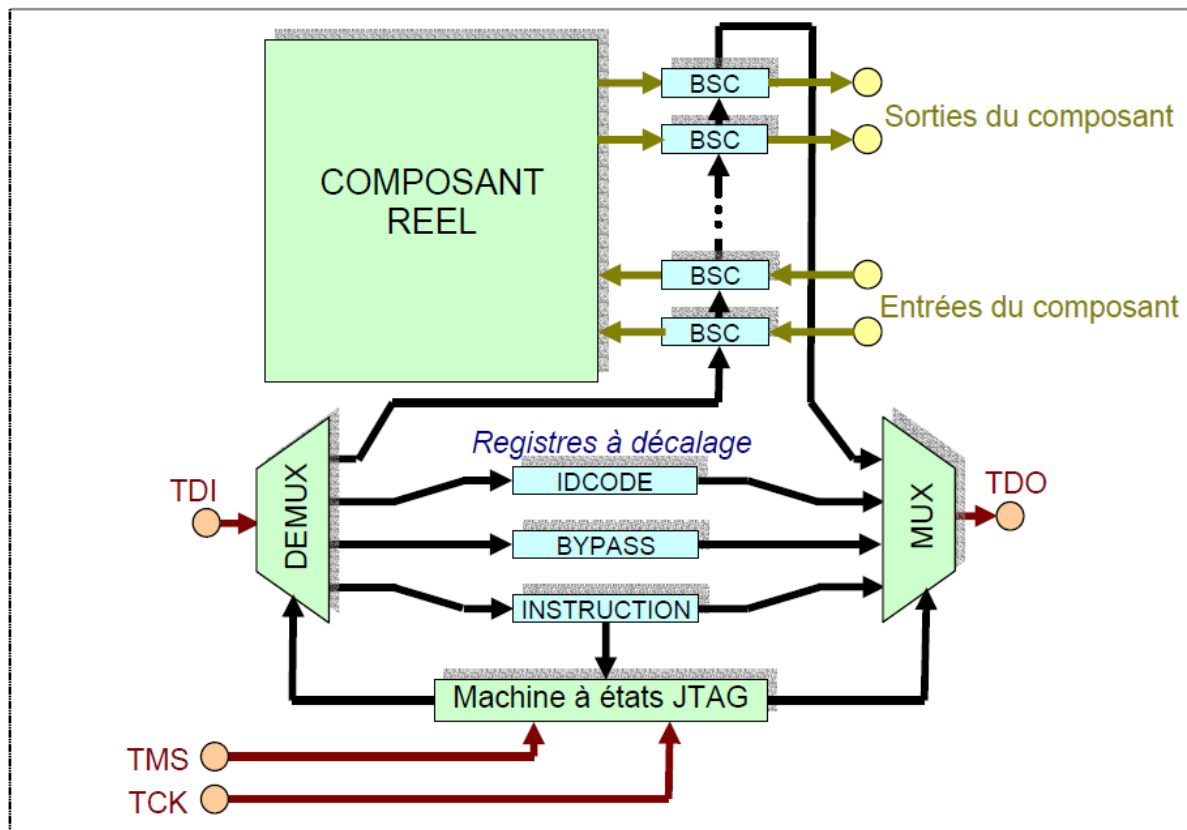


Figure 5 : Synoptique de l'interface JTAG d'un composant.

L'interface dispose de registres à décalage (*en bleu*) dans lesquels les informations caractéristiques du circuit sont stockées. Les informations qui sont alors sérialisées en entrée, ressortent au fur et à mesure en sortie lorsque le registre est plein. Ainsi les informations supplémentaires sont véhiculées vers le composant suivant sur la chaîne.

Le coeur de cette interface est la « Machine à état JTAG », que l'on appelle aussi contrôleur « TAP » (Test Access Port). Suivant ce qui se trouve dans le registre d'instruction, ce contrôleur aiguille les informations sur le registre de travail. La ligne TMS, cadencée avec TCK, permet de choisir la condition de la machine JTAG (dont le principe est expliqué plus loin). Un registre particulier, le Boundary-Scan (BSC) fait correspondre des données aux états des entrées et sorties du composant réel.

### 3- La machine JTAG

Tous les composants compatibles à la norme JTAG disposent d'un contrôleur pour l'interface. Ce système de contrôle est appelé « Machine à états JTAG ». En effet, il fait basculer l'interface dans un régime de fonctionnement donné (état), en suivant des règles de progression prédéfinies, afin d'épouser le fonctionnement des circuits.

Relativement à l'interface, les informations véhiculées par les lignes TDI / TDO, peuvent être de deux type :

- **Les données**, qui sont des informations brutes.
- **Les instructions**, qui concernent le fonctionnement de l'interface JTAG.

Les données sont des informations brutes concernant l'architecture du composant réel, tandis que les instructions sont des mots qui définissent l'opération que l'interface doit réaliser.

La *figure 7* de la page suivante, montre le diagramme de transition entre les différents états du bus JTAG, et est utile à la compréhension de son fonctionnement. La machine est constituée de différents états qui sont conditionnés par l'envoi de 0 ou de 1 logiques sur la ligne TMS.

Dans ce diagramme il existe deux colonnes identiques qui sont :

- Colonne d'instruction IR (Instruction Register)  
Cette partie du diagramme permet de configurer le registre d'instruction qui permet de choisir le mode de fonctionnement du Jtag
- Colonne de données DR (Data Register)  
Cette partie du diagramme permet de charger les registres de données selon le mode de fonctionnement choisi, ce mode est déterminé par le registre d'instruction.

Sur le diagramme suivant, on peut remarquer que pour arriver à un état donné, il y a quelquefois plusieurs chemins possibles. Certains sont plus courts que d'autres car ils permettent d'optimiser la durée des tests. Toutefois, il est primordial de revenir à l'état IDLE pour que le circuit réalise l'action si elle est prête à être exécutée.

### ***3 - 1 Diagramme de fonctionnement***

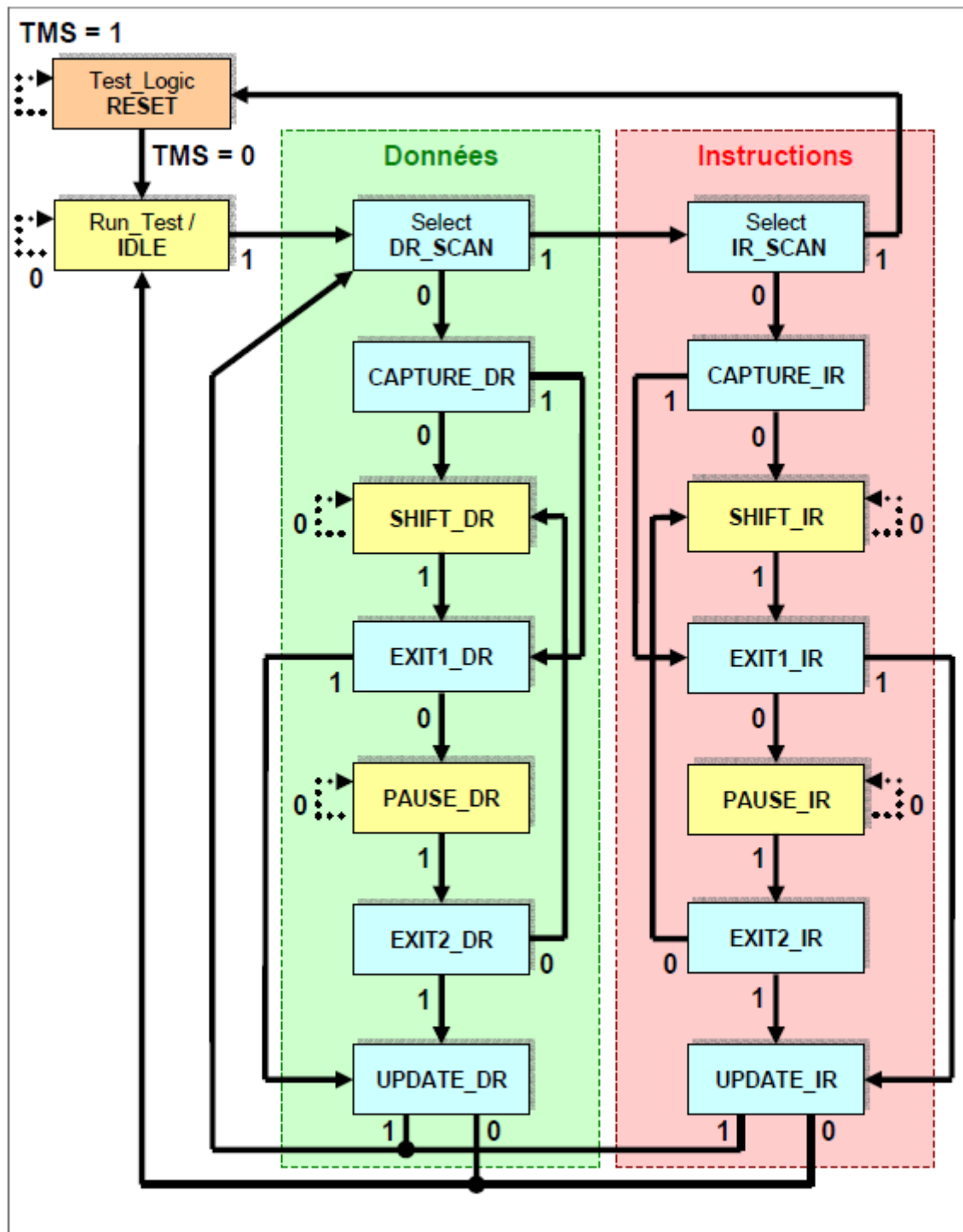


Figure 7 : Digramme de fonctionnement de la Machine JTAG.

### 3 – 2 Les états du diagramme

- Test-Logic-Reset : état de reset
- Run-Test-Idle : état de mise en attente
- Select-DR/IR : état temporaire pour choisir le registre d'instruction/donnée ou reset.
- Capture-IR/DR : Le registre à décalage IR/DR est chargé en parallèle sur front montant de TCK.

- Shift-IR/DR : Les valeurs sont décalées de l'entrée TDI vers la sortie TDO dans le registre à décalage IR/DR sur front montant de TCK.
- Exit1-IR/DR : état temporaire
- Pause-IR/DR : état de pause (attente de valeurs sur TDI)
- Exit2-IR/DR : état temporaire
- Update-IR/DR : les valeurs contenues dans le registre à décalage IR/DR sont chargées dans le registre parallèle IR/DR sur front descendant de TCK. prise en compte des nouvelles données par le circuit.

## 4 - Les jeux d'instructions

Une instruction est un mot binaire d'une longueur propre à chaque constructeur (ex : pour *Altera*, les instructions sont d'une longueur de 10 bits, tandis qu'ils sont de 8 bits chez *Actel*) et sont définies dans les fichiers de description des composants (BSDL) fournis par les fabricants.

Pour chaque type de transaction, il y a un jeu d'instructions standardisé qui permet de réaliser les actions souhaitées. Les instructions impératives sont communes à chaque composant, tandis que les instructions optionnelles sont différentes d'un constructeur à l'autre. **Les instructions EXTEST, BYPASS, et SAMPLE/PRELOAD (Imposées par la norme IEEE 1149.1)** sont définies de façon à ne pas dépendre de la longueur des registres (le minimum étant de 8 bits).



<i>Instruction</i>	<i>Code (ex. 10 bits)</i>	<i>Registre sélectionné</i>	<i>Action effectuée</i>
<b>Instructions impératives</b>			
EXTEST	00'0000'0000	Boundary-Scan	Lecture et/ou écriture sur les entrées et sorties du composant virtuel*
SAMPLE/ PRELOAD	00'0101'0101	Boundary-Scan	Lecture et/ou écriture du registre Boundary-Scan.
BYPASS	11'1111'1111	Bypass	Répéter les informations venant de l'entrée.
<b>Instructions optionnelles</b>			
INTEST	-	Boundary-Scan	Lecture et/ou écriture directe sur les entrées et sorties du composant réel*.
RUNBIST	-	(Aucun)	Exécution d'un test autonome intégré (Built In Autotest) du vrai composant.
IDCODE	-	Identifiant constructeur	Requête du code d'identification propre du composant.
USERCODE	-	Identifiant utilisateur	Requête du code d'identification choisit par l'utilisateur pour un composant.
CLAMP	-	Bypass	Maintien de l'état des entrées et sorties du composant virtuel*, puis répéter les informations venant de l'entrée.
HIGH-Z	-	Bypass	Toutes les sorties du composant sont mises à l'état de haute impédance.

*Instructions conventionnelles du contrôleur JTAG*

\* Le composant virtuel est celui que l'on observe à la sortie des cellules JTAG (qui englobent les entrées et sorties). Le composant réel est celui qui n'est pas affecté par les cellules JTAG.

## 5 - Mode opératoire

Partant de l'état IDLE, le mode opératoire classique consiste à déposer l'instruction en SHIFT\_IR, et de récupérer les informations en SHIFT\_DR en passant par les états intermédiaires. Lorsque plusieurs circuits sont montés sur la chaîne JTAG, les instructions sont déposées les unes après les autres dans l'ordre de montage par rapport à la source, et tous les circuits se partagent la chaîne d'instructions. Elles sont ensuite validées en sortant du mode SHIFT\_IR et chaque composant se retrouve avec son instruction propre.

### 5 - 1 Exemple de requête d'identifiant constructeur (IDCODE)

Afin de mieux comprendre comment on manipule les signaux sur les lignes JTAG, on analyse l'oscillogramme visualisé sur l'oscilloscope numérique (*Figure 11*). Il s'agit d'un test réalisé avec 2 circuits Altera chaînés par le bus JTAG sur un prototype du circuit de développement. Le premier composant est une mémoire flash (référence EPC2L20), et le deuxième est un circuit programmable FPGA (ACEX EP1K100) (*Figure 8*).

On visualise à l'oscilloscope l'évolution des niveaux des 4 lignes de l'interface (TDI,

TDO, TMS et TCK), et la ligne intermédiaire pour observer les échanges entre les 2 circuits (TDO2).

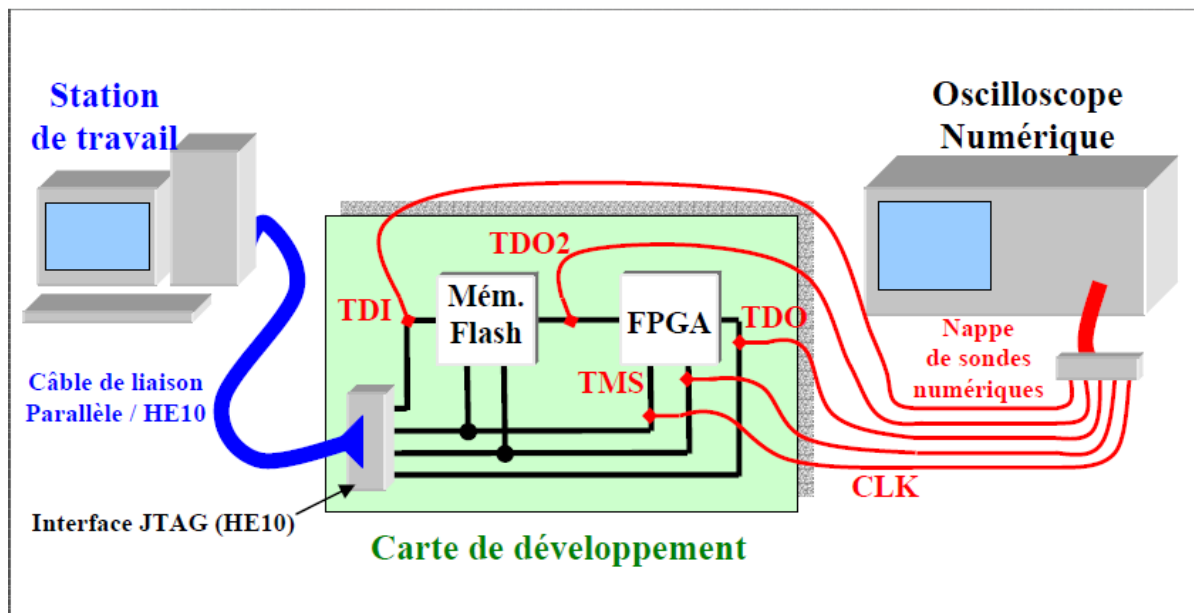


Figure 8 : Schéma du montage de l'expérience.

Dans cet expérience, on va demander uniquement le numéro d'identifiant du deuxième circuit sur la chaîne. Etant donné que les circuits sont montés en cascade, la difficulté est que le premier circuit ne doit pas répondre pendant que le code d'identification est lu à la sortie. Il faut donc que l'on attribue le mode **BYPASS** au premier composant et le mode **IDCODE** au second.

Pour le constructeur *Altera*, les instructions sont des mots de 10 bits qui sont donc communs aux 2 composants (Figure 9).

Fonction	Instruction pour le FPGA	Instruction pour la Flash
IDCODE (Optionnel)	00'0000'0110	00'0101'1001
BYPASS (Standard)	11'1111'1111	11'1111'1111

Figure 9 : Instructions correspondantes aux fonctions.

Le code pour l'instruction **IDCODE** est fourni par les documentations *Altera* et le fichier **BSDL** qui correspond au composant. Comme il s'agit d'une instruction optionnelle, elle est différente pour les deux composants.

Les numéros d'identifiants pour les deux circuits (Figure 10) sont des clefs de 32 bits qui sont standardisées pour tout composant à la norme *IEEE 1149.1* :

Composant	Clef d'identification du constructeur (32 bits)			
	Version (4 bits)	Circuit (16 bits)	Fabricant (11 bits)	1
Flash, EPC2L20	0000	0001'0000'0000'0010	0000'1101'110	1
FPGA, ACEX EP1K100	0010	0000'0001'0000'0000	0000'1101'110	1

Figure 10 : Numéros d'identifiant pour les 2 composants.

La première partie de cette clef, est la version du composant (pour la flash, version 0 signifie qu'il n'y aura pas d'autre version). Ensuite, c'est le numéro propre au composant, qui est unique pour chaque constructeur. Le numéro de fabricant est celui que *Altera* s'est approprié, c'est pourquoi il est identique pour les deux composants. Le dernier bit est toujours à 1 pour permettre de vérifier qu'il s'agit bien d'une clef constructeur.

**Procédure à suivre :** dans un premier temps, il faut attribuer à chaque circuit la fonction que l'on souhaite lui faire réaliser (instruction), et récupérer les informations à la sortie (clef d'identification).

L'oscillogramme du test est présenté par la Figure 11.

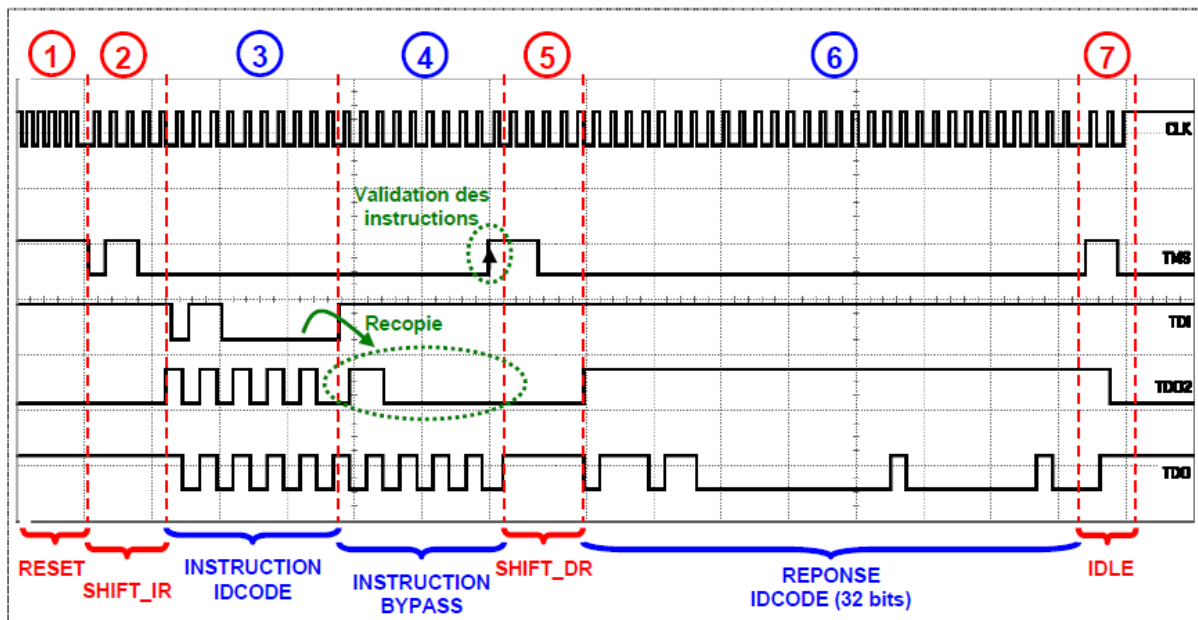


Figure 11 : Oscillogramme d'une procédure de requête IDCODE.

La première des opérations est de réinitialiser la machine JTAG à l'état RESET (1). Grâce à la structure de la machine à états, en envoyant cinq niveaux haut sur TMS (soit « 11111 »), on arrive à l'état RESET quel que soit l'état précédant.

Ensuite, on sélectionne le registre des instructions pour déclarer l'action que l'on veut réaliser. On se positionne dans l'état SHIFT\_IR par l'écriture des bits « 01100 » sur TMS (2). Après cela on peut envoyer les instructions, tant que TMS reste à 0 (état stable).

Les instructions sont envoyées dans l'ordre inverse de la disposition des composants sur la chaîne JTAG : du dernier au premier. Ici, on souhaite obtenir le numéro d'identifiant du dernier composant de la chaîne. Il faut donc que le premier composant soit en mode transparent BYPASS pour qu'il ne réponde pas à l'instruction IDCODE. On enchaîne donc l'instruction IDCODE (3) avec BYPASS (4) sur la ligne TDI. On remarque que dans le mode

SHIFT\_IR, tant que le registre n'a pas reçu au moins 10 bits, le composant répond par une alternance de 1 et de 0. Ensuite, il recopie les 10 premiers bits reçus (4) (l'instruction destinée au circuit suivant, qui est visible sur la ligne TDO2). Ainsi, lorsque l'on valide, par un front montant de TMS, sur le dernier bit des instructions, les 10 derniers bits sont inscrits dans les registres d'instructions. Cette validation positionne la machine à état en EXIT1\_IR.

Les instructions ne sont pas encore exécutées, il faut aller à l'état SHIFT\_DR par l'écriture des bits « 1100 » sur TMS (5). Alors, aux 32 fronts descendants suivant, on récupère les 32 bits du numéro d'identifiant constructeur (6). On constate que le numéro correspond bien à celui attendu du FPGA : c'est donc celui-ci qui a répondu. On observe que le premier circuit répète bien les informations qui lui parviennent à l'entrée, puisque la ligne TDO2 est au même niveau que TDI dès que l'on rentre dans le mode SHIFT\_DR.

Enfin, pour que l'interface soit prête à effectuer une autre opération, on se repositionne dans l'état de repos IDLE par l'écriture des bits « 110 » sur TMS (7).

## Résumé :

- L'interface JTAG se présente comme une solution incontestable de test des composants. Il est alors possible de déterminer les circuits défectueux et les pistes en court-circuit ou altérées.
- La transmission des données parcourant les composants se fait de façon série.
- Le cœur de cette interface est la machine JTAG qui aiguille les informations sur le registre de travail.